Title: LINE CACHE CONTROLLER
Inventor: Ho, Son et al.
Atty. Ref. No.: MP0390

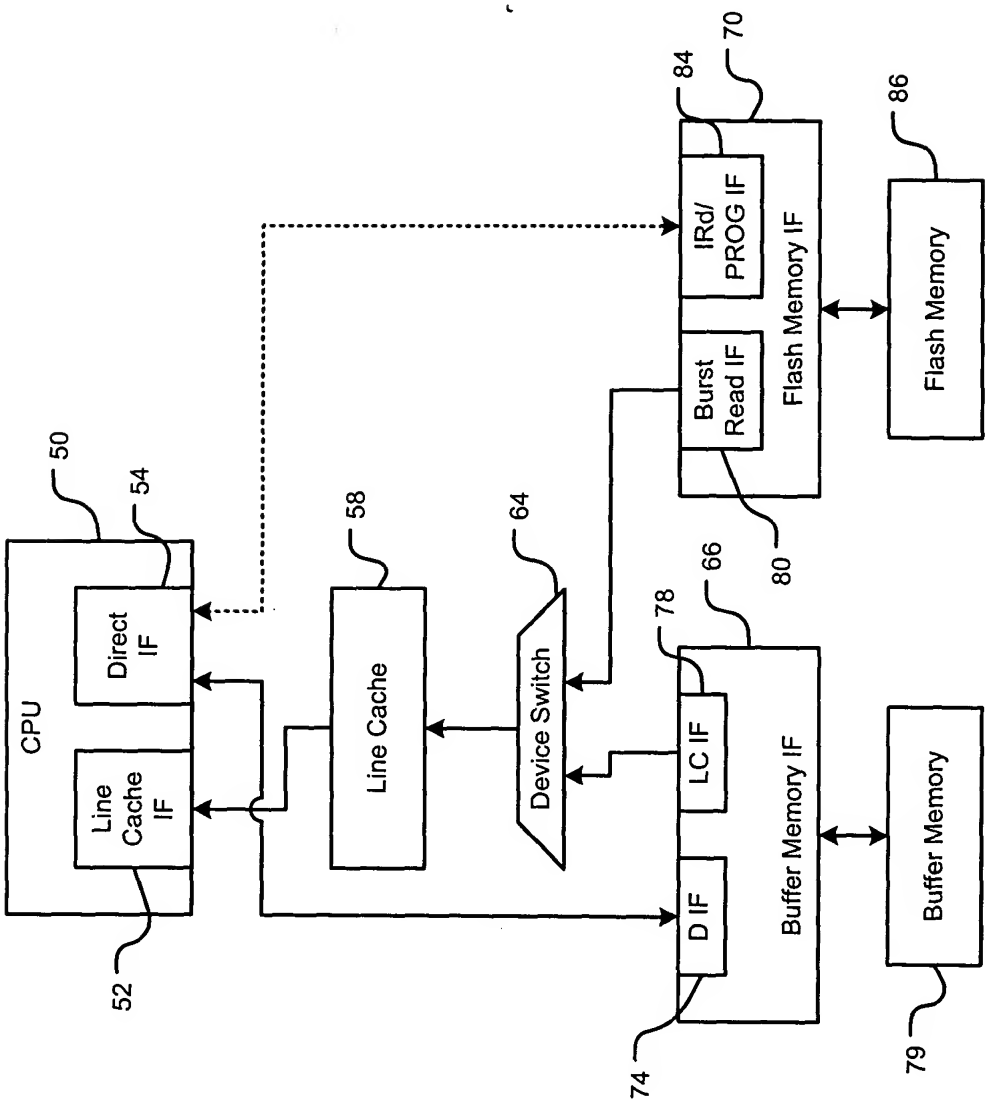

FIG.1

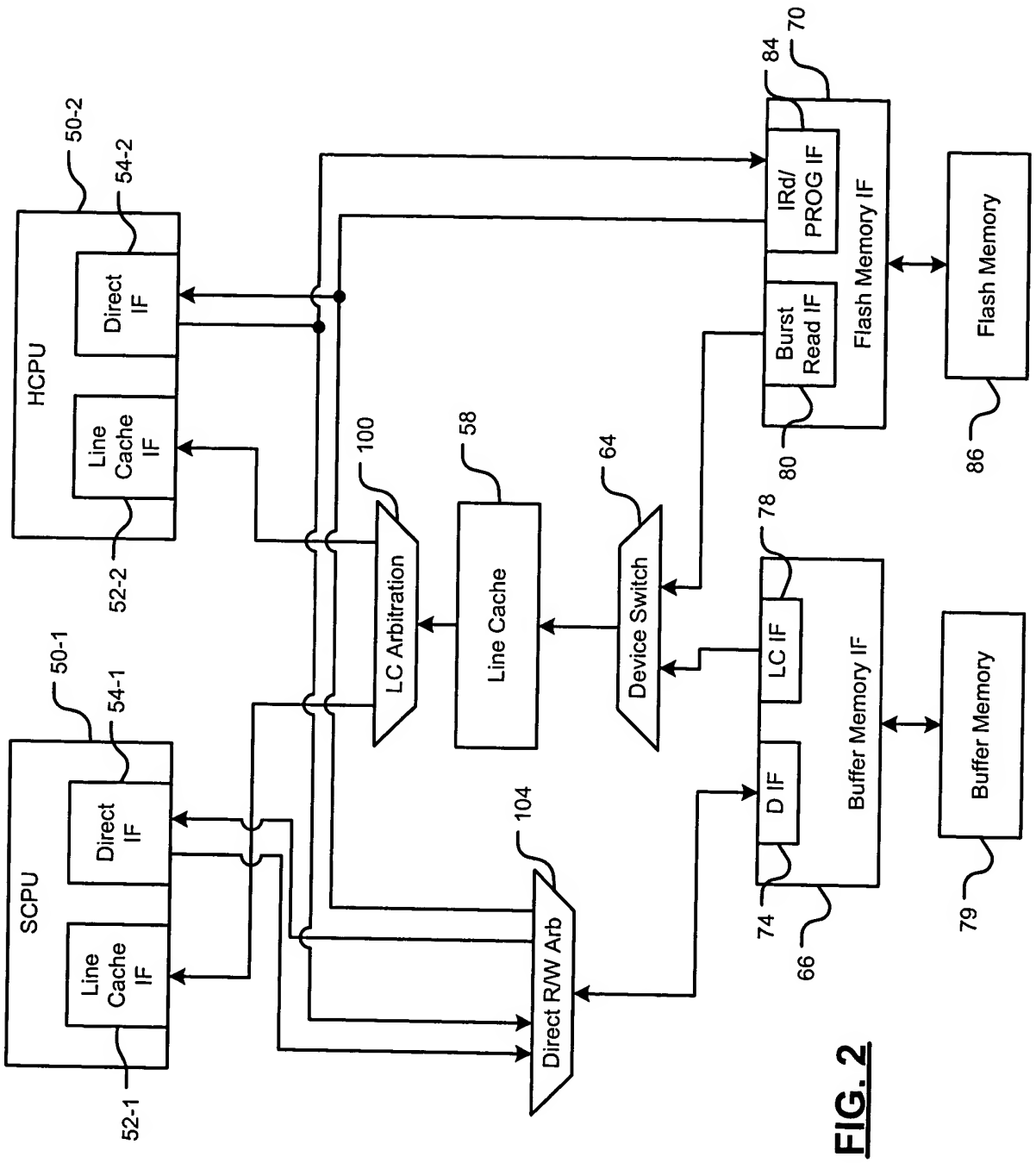
HCPU
Direct IF
Line Cache IF
SCPU
Direct IF
Line Cache IF
LC Arbitration
Line Cache
Device Switch
Direct R/W Arb
IRd/ PROG IF
Burst Read IF
Flash Memory IF
Flash Memory
LC IF
D IF
Buffer Memory IF
Buffer Memory
50-2
54-2
52-2
50-1
54-1
52-1
100
58
64
104
84
70
80
86
78
74
66
79

FIG. 2

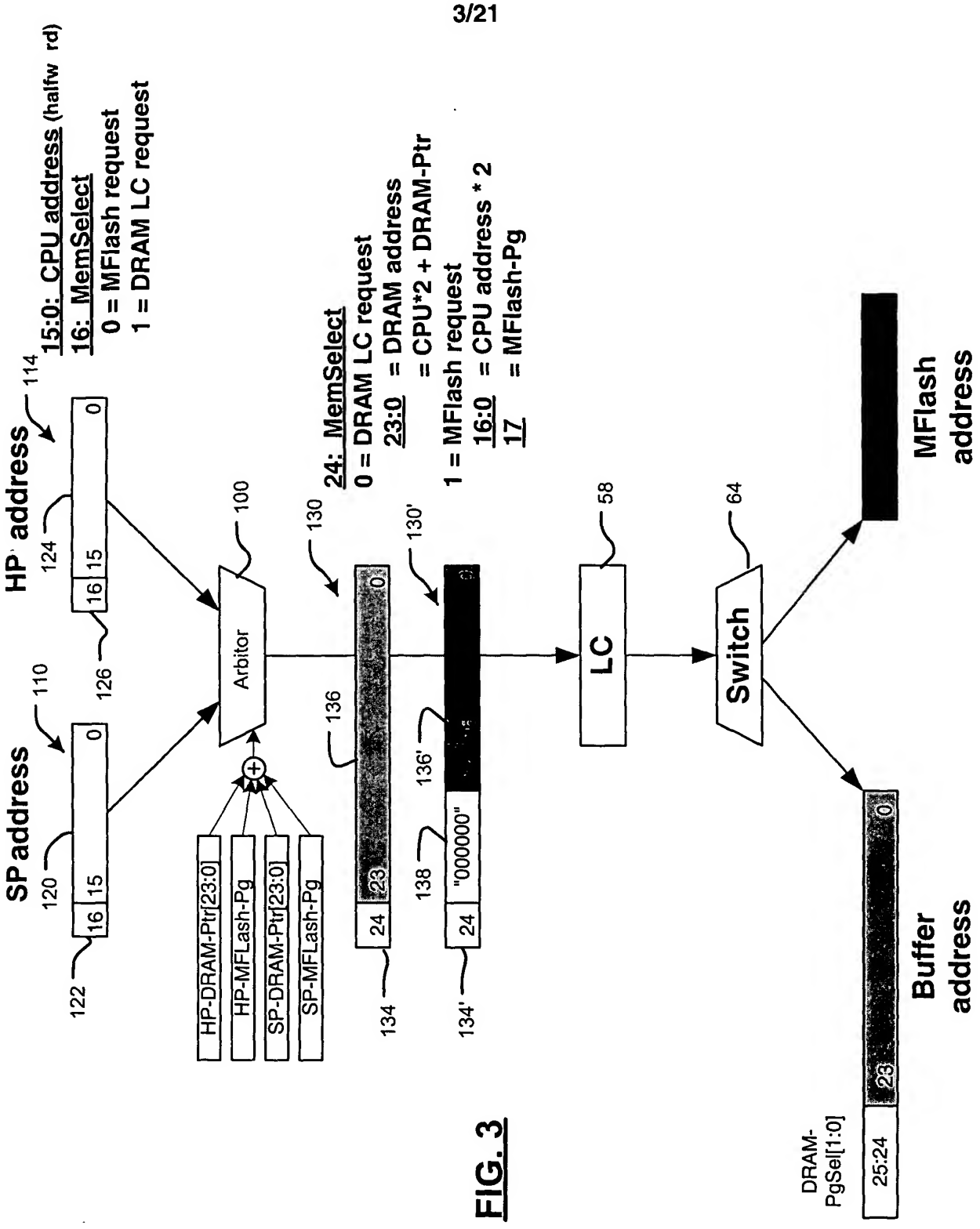
SP address
HP address
15:0: CPU address (halfw rd)
16: MemSelect
0 = MFlash request
1 = DRAM LC request
24: MemSelect
0 = DRAM LC request
23:0 = DRAM address
= CPU*2 + DRAM-Ptr
1 = MFlash request
16:0 = CPU address * 2
17 = MFlash-Pg
Arbitor
HP-DRAM-Ptr[23:0]
HP-MFLash-Pg
SP-DRAM-Ptr[23:0]
SP-MFLash-Pg
"000000"
LC
Switch
MFlash address
Buffer address
DRAM-PgSel[1:0]
25:24
100
110
114
120
122
124
126
130
130'
134
134'
136
136'
138
58
64

FIG. 3

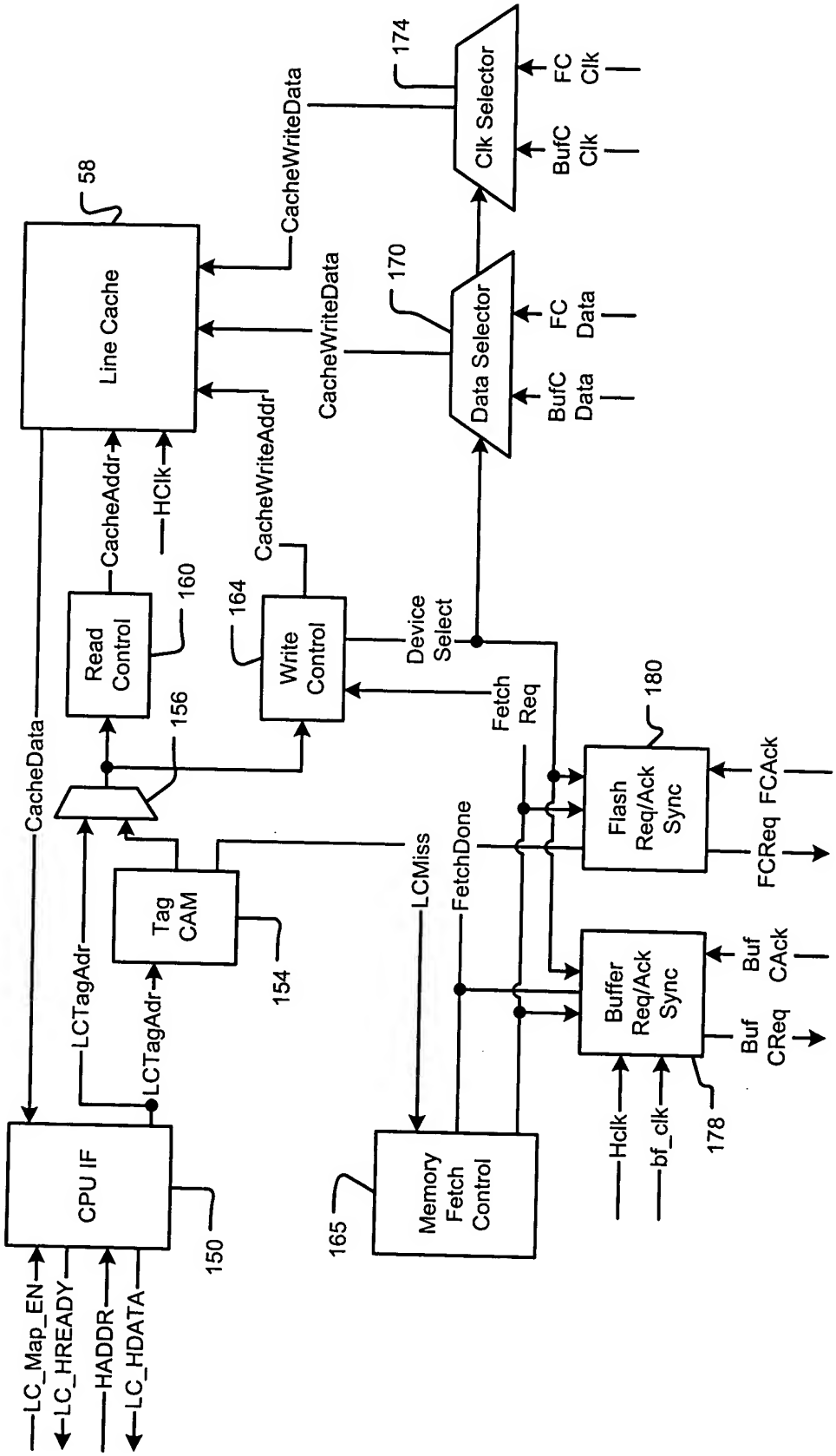
Line Cache
58
CacheWriteData
CacheWriteData
CacheWriteAddr
CacheAddr
HClk
CacheData
Clk Selector
174
FC Clk
BufC Clk
Data Selector
170
FC Data
BufC Data
Read Control
160
Write Control
164
Device Select
Fetch Req
156
Tag CAM
154
LCTagAdr
LCTagAdr
LCMiss
FetchDone
Flash Req/Ack Sync
180
FCReq FCAck
Buffer Req/Ack Sync
Buf CReq
Buf CAck
Hclk
bf_clk
178
CPU IF
150
Memory Fetch Control
165
LC_Map_EN
LC_HREADY
HADDR
LC_HDATA

FIG. 4

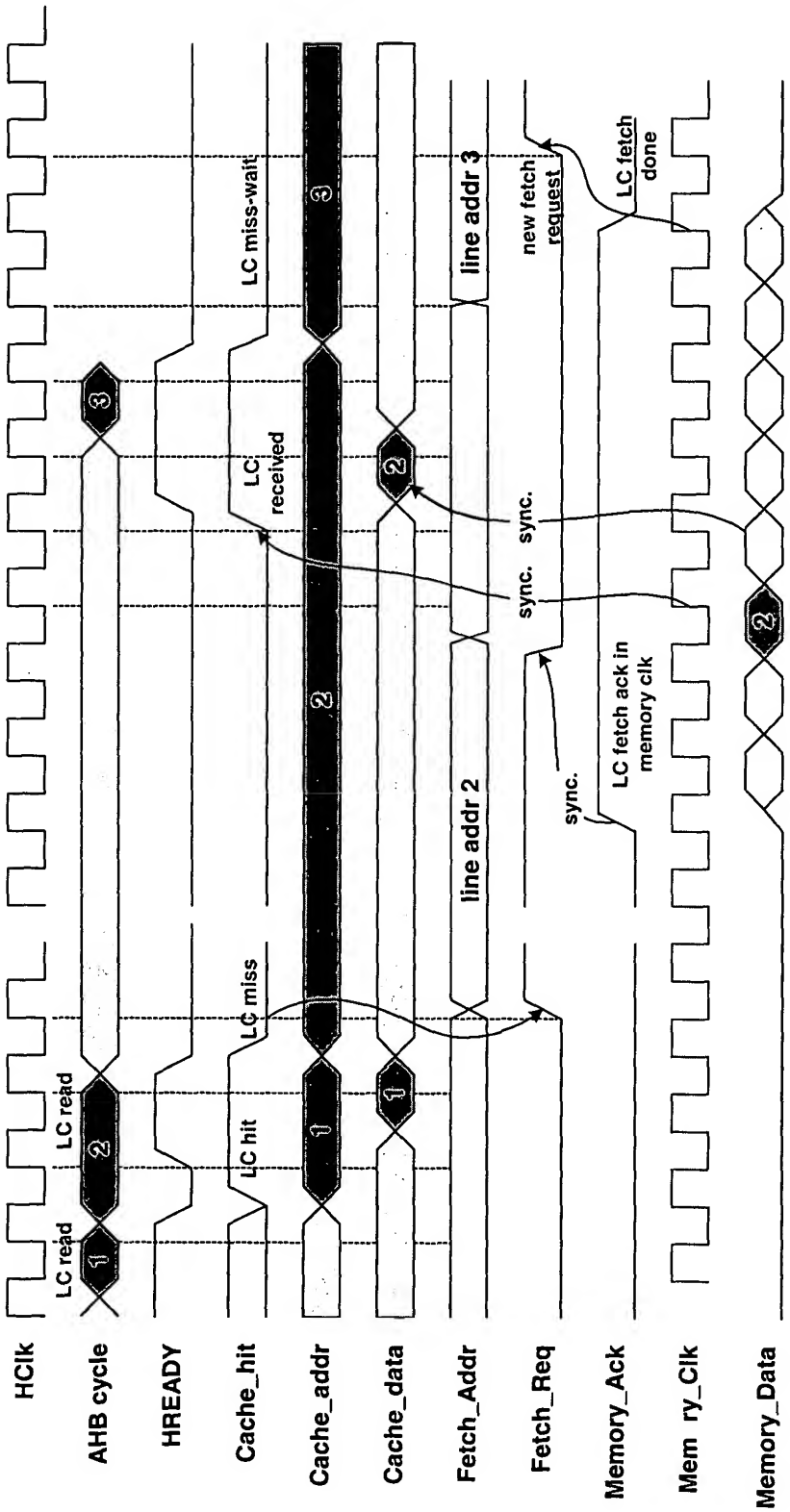
HClk
AHB cycle
HREADY
Cache_hit
Cache_addr
Cache_data
Fetch_Addr
Fetch_Req
Memory_Ack
Mem ry_Clk
Memory_Data
LC read
LC read
LC hit
LC miss
LC miss-wait
LC received
line addr 2
line addr 3
sync.
sync.
sync.
new fetch request
LC fetch ack in memory clk
LC fetch done
1
2
3

FIG. 5

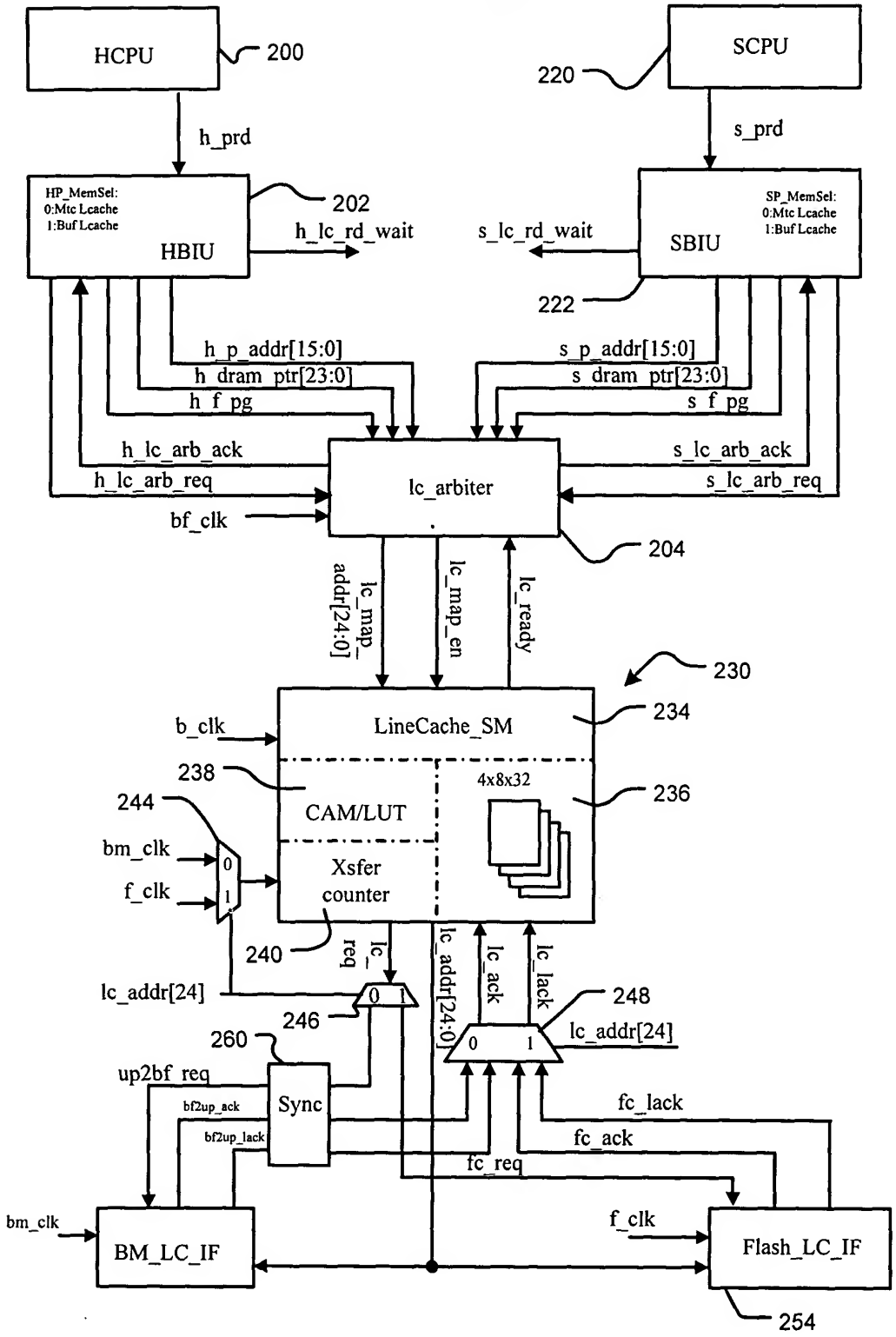
HCPU
200
SCPU
220
h_prd
s_prd
HP_MemSel:
0:Mtc Lcache
1:Buf Lcache
HBIU
202
h_lc_rd_wait
s_lc_rd_wait
SP_MemSel:
0:Mtc Lcache
1:Buf Lcache
SBIU
222
h_p_addr[15:0]
h_dram_ptr[23:0]
h_f_pg
s_p_addr[15:0]
s_dram_ptr[23:0]
s_f_pg
h_lc_arb_ack
h_lc_arb_req
bf_clk
lc_arbiter
s_lc_arb_ack
s_lc_arb_req
204
lc_map_addr[24:0]
lc_map_en
lc_ready
230
b_clk
LineCache_SM
234
238
CAM/LUT
4x8x32
236
244
bm_clk
f_clk
Xsfer
counter
240
lc_req
lc_addr[24]
246
lc_addr[24:0]
lc_ack
lc_lack
248
260
up2bf_req
Sync
bf2up_ack
bf2up_lack
fc_lack
fc_ack
fc_req
bm_clk
BM_LC_IF
f_clk
Flash_LC_IF
254

FIG. 6

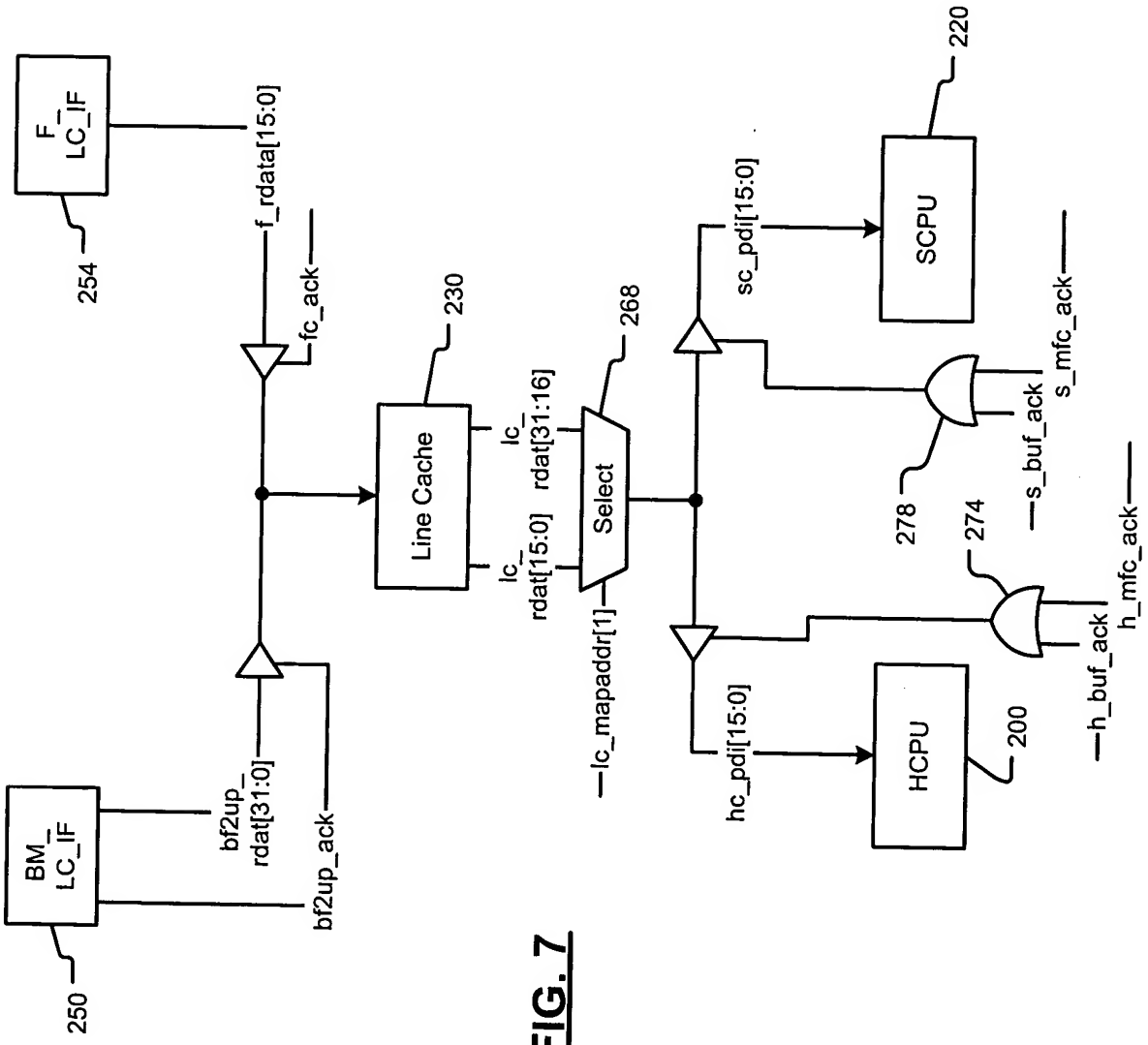
F_ LC_IF
254
f_rdata[15:0]
fc_ack
BM_ LC_IF
250
bf2up_ rdat[31:0]
bf2up_ack
Line Cache
230
lc_ rdat[31:16]
lc_ rdat[15:0]
Select
268
lc_mapaddr[1]
sc_pdi[15:0]
SCPU
220
hc_pdi[15:0]
HCPU
200
278
274
s_buf_ack
s_mfc_ack
h_buf_ack
h_mfc_ack

FIG. 7

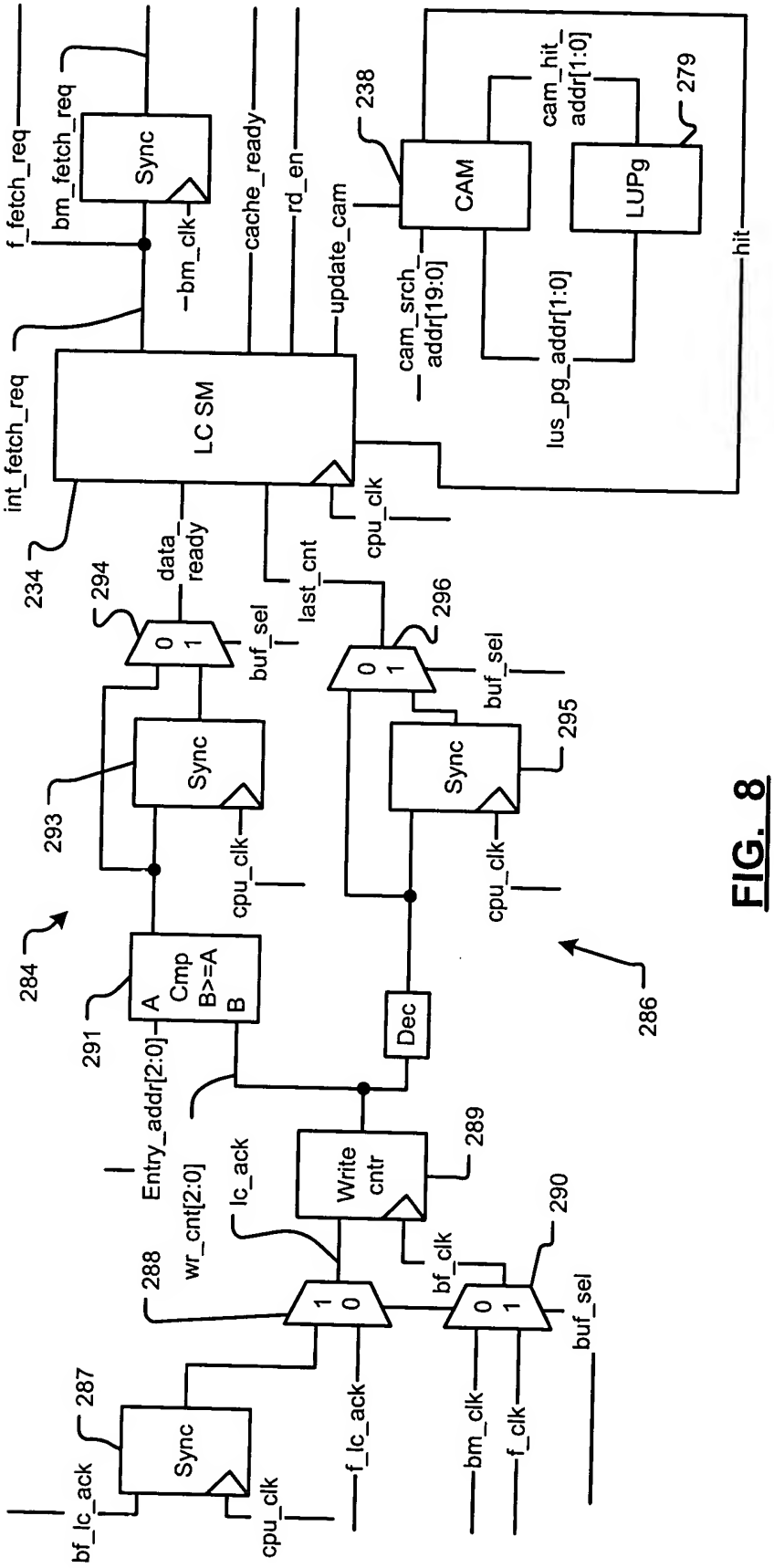
f_fetch_req
bm_fetch_req
Sync
bm_clk
cache_ready
rd_en
update_cam
238
CAM
cam_hit_addr[1:0]
279
LUPg
hit
cam_srch_addr[19:0]
lus_pg_addr[1:0]
int_fetch_req
LC SM
cpu_clk
234
data_ready
294
last_cnt
296
buf_sel
0
1
Sync
293
cpu_clk
Sync
295
284
A
Cmp
B>=A
B
291
Entry_addr[2:0]
Dec
286
wr_cnt[2:0]
lc_ack
Write cntr
289
288
bf_clk
290
buf_sel
bf_lc_ack
287
Sync
cpu_clk
f_lc_ack
bm_clk
f_clk

FIG. 8

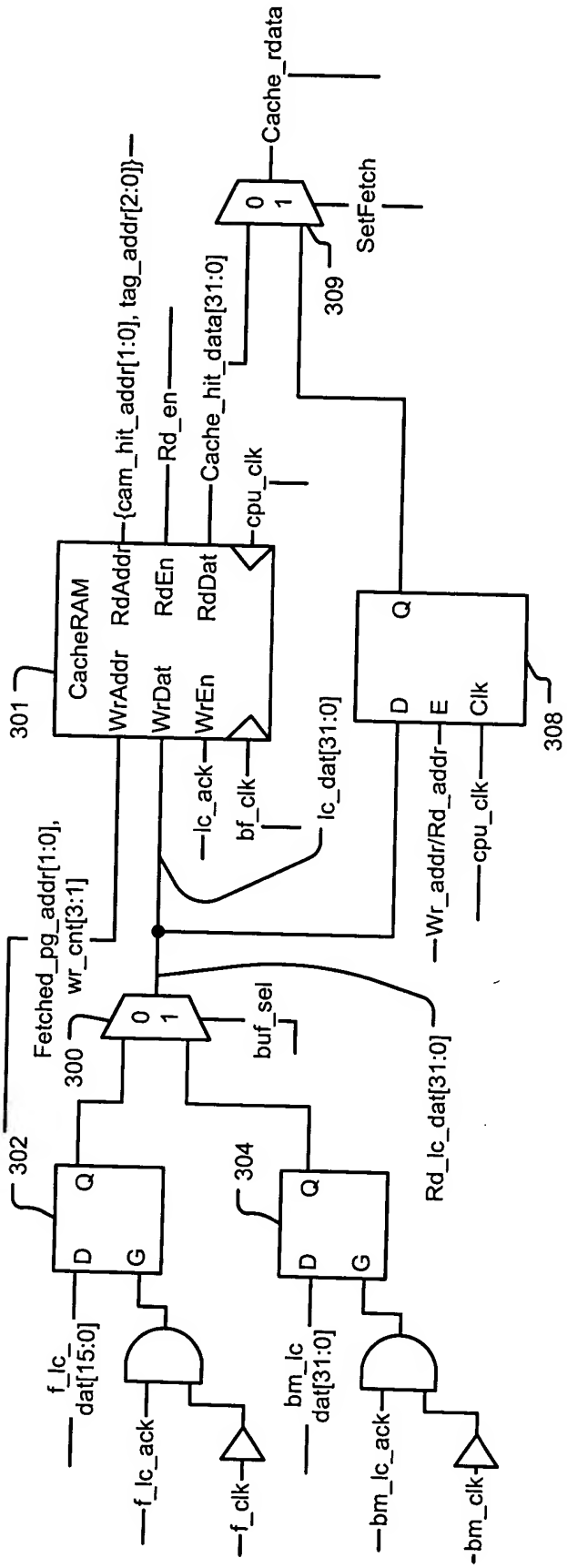
Cache_rdata
SetFetch
309
{cam_hit_addr[1:0], tag_addr[2:0]}
Rd_en
Cache_hit_data[31:0]
cpu_clk
301
CacheRAM
WrAddr
RdAddr
WrDat
RdEn
WrEn
RdDat
lc_ack
bf_clk
lc_dat[31:0]
308
Q
D
E
Clk
Wr_addr/Rd_addr
cpu_clk
Fetched_pg_addr[1:0],
wr_cnt[3:1]
300
buf_sel
Rd_lc_dat[31:0]
302
304
f_lc_
dat[15:0]
f_lc_ack
f_clk
bm_lc
dat[31:0]
bm_lc_ack
bm_clk

FIG. 9

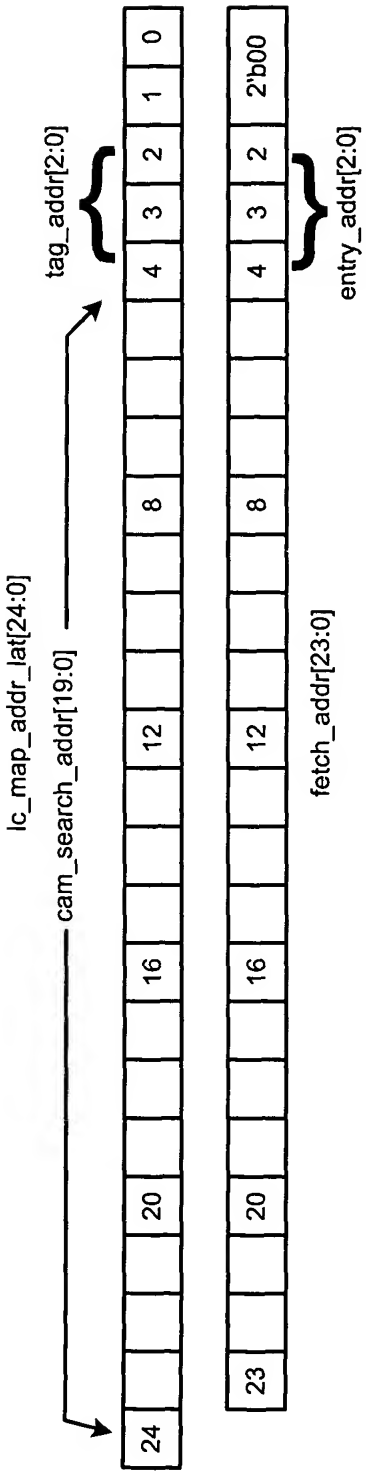

FIG. 10

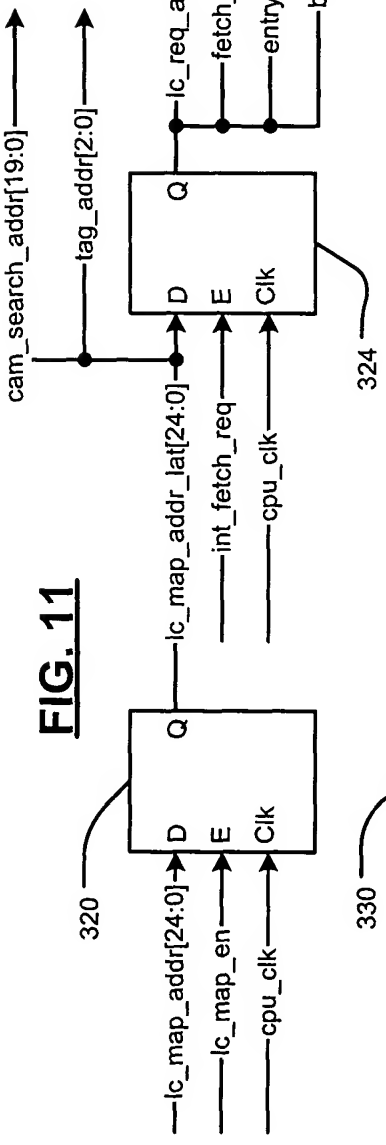

FIG. 11

FIG. 12

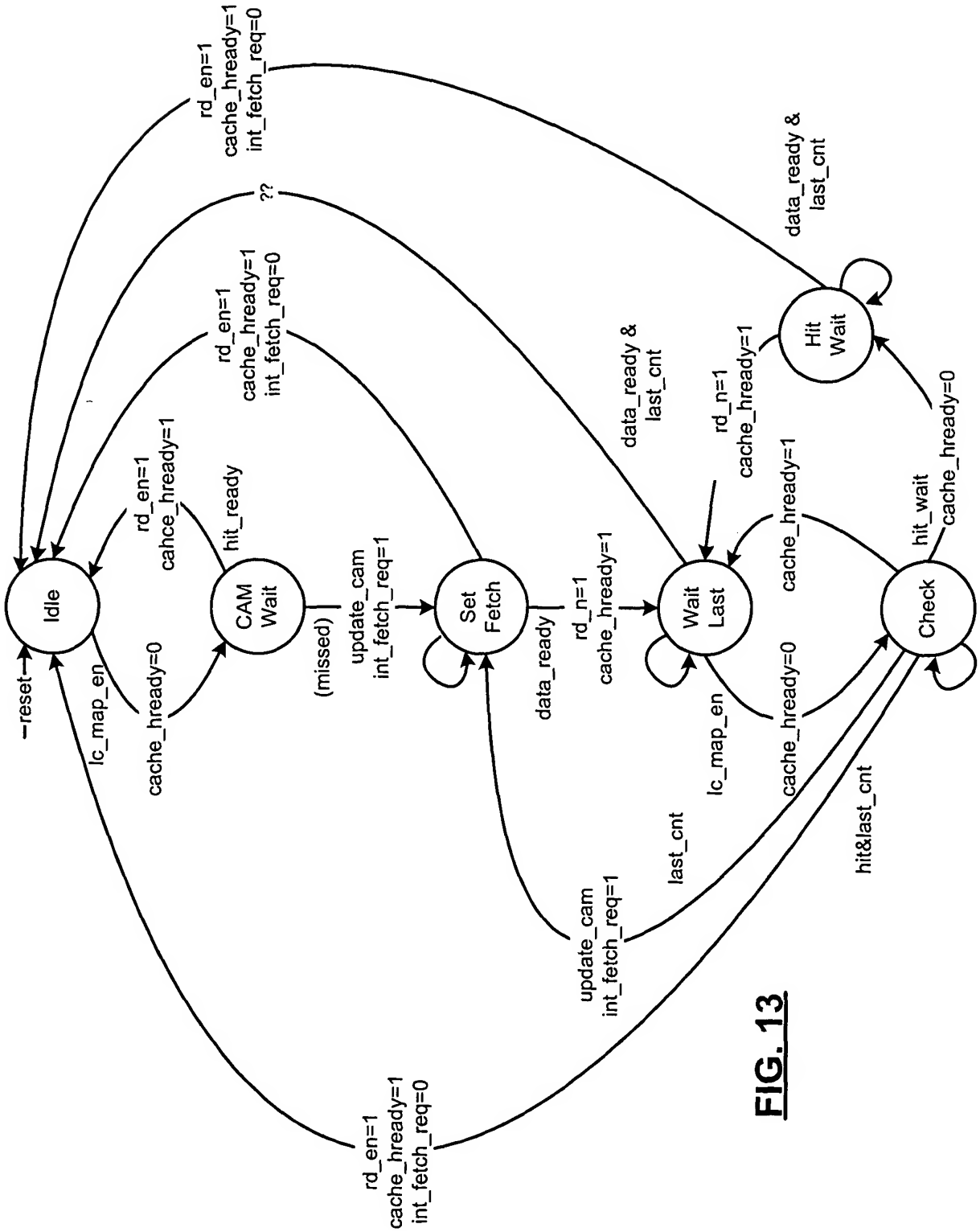
Idle
CAM Wait
Set Fetch
Wait Last
Check
Hit Wait
−reset
lc_map_en
cache_hready=0
rd_en=1
cahce_hready=1
hit_ready
(missed)
update_cam
int_fetch_req=1
data_ready
rd_n=1
cache_hready=1
last_cnt
hit_wait
rd_en=1
cache_hready=1
int_fetch_req=0
??
data_ready &
last_cnt
hit&last_cnt

FIG. 13

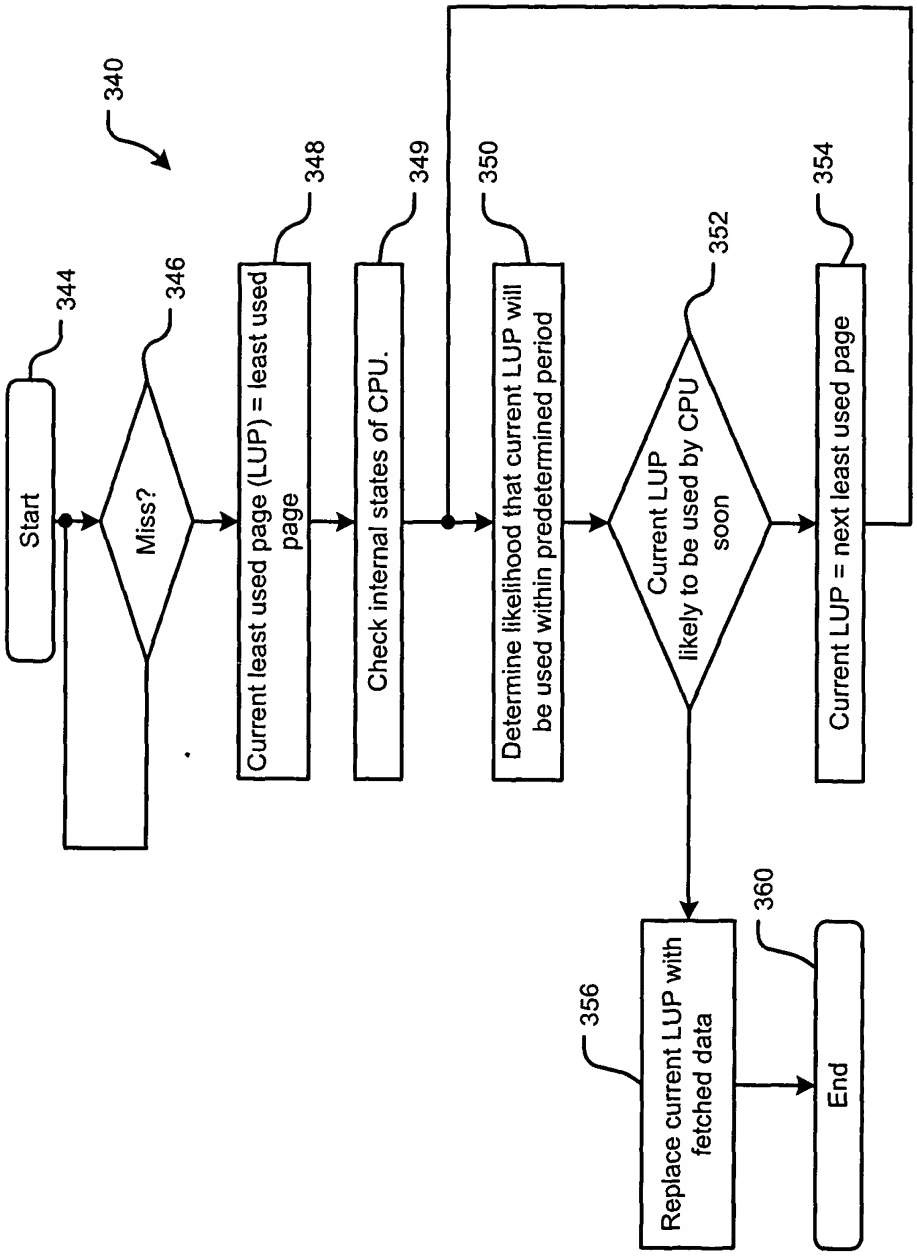
340
344
Start
346
Miss?
348
Current least used page (LUP) = least used page
349
Check internal states of CPU.
350
Determine likelihood that current LUP will be used within predetermined period
352
Current LUP likely to be used by CPU soon
354
Current LUP = next least used page
356
Replace current LUP with fetched data
360
End

FIG. 14

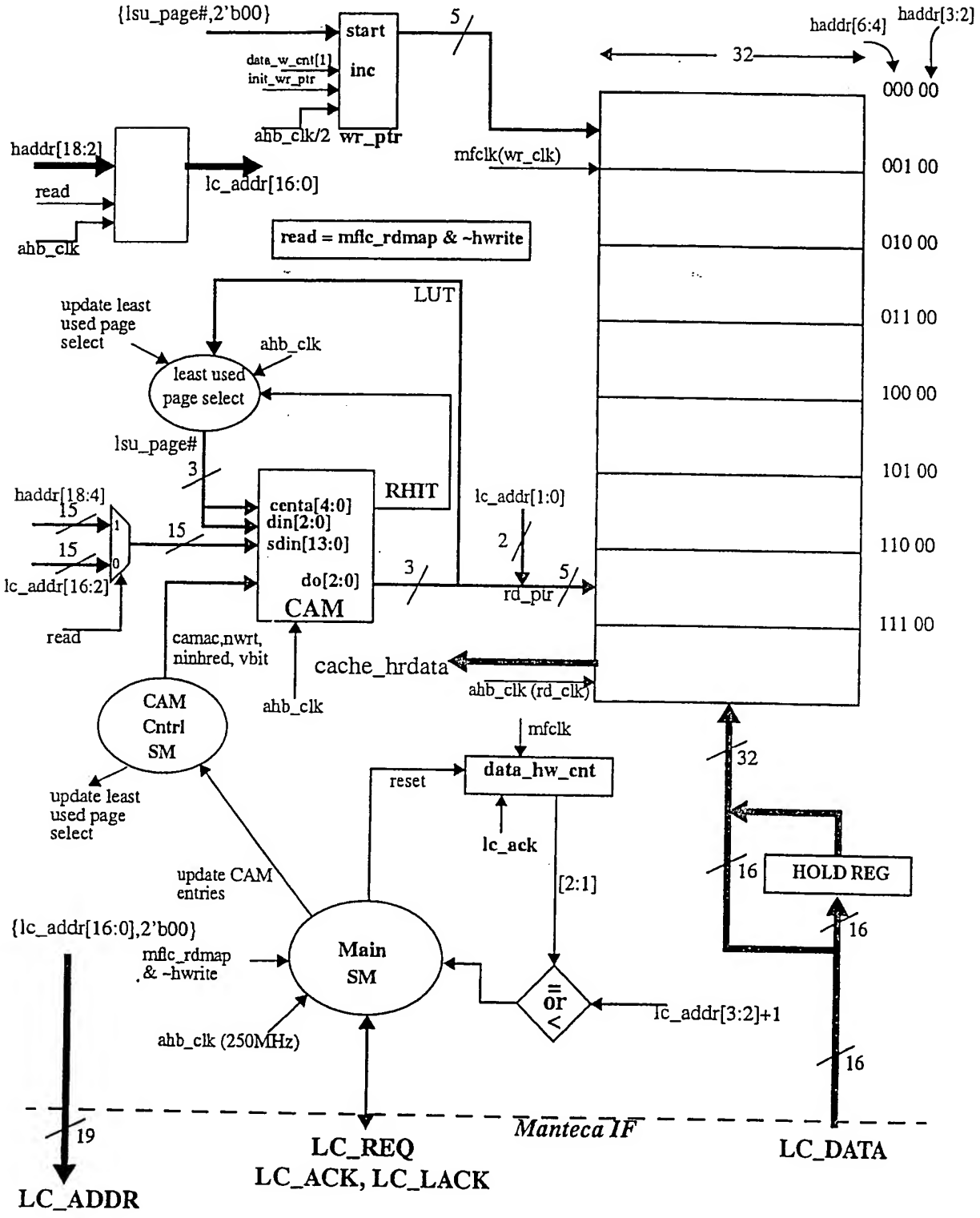
{lsu_page#,2'b00}
start
inc
data_w_cnt[1]
init_wr_ptr
ahb_clk/2
wr_ptr
5
haddr[6:4]
haddr[3:2]
32
000 00
001 00
010 00
011 00
100 00
101 00
110 00
111 00
mfclk(wr_clk)
haddr[18:2]
read
ahb_clk
lc_addr[16:0]
read = mfic_rdmap & ~hwrite
LUT
update least used page select
ahb_clk
least used page select
lsu_page#
3
haddr[18:4]
15
15
15
lc_addr[16:2]
read
centa[4:0]
din[2:0]
sdin[13:0]
do[2:0]
CAM
RHIT
lc_addr[1:0]
2
3
5
rd_ptr
camac,nwrt, ninhred, vbit
ahb_clk
cache_hrdata
ahb_clk (rd_clk)
CAM Cntrl SM
update least used page select
mfclk
data_hw_cnt
reset
lc_ack
32
16
HOLD REG
16
update CAM entries
[2:1]
{lc_addr[16:0],2'b00}
mfic_rdmap .& ~hwrite
Main SM
ahb_clk (250MHz)
=
or
<
lc_addr[3:2]+1
16
19
Manteca IF
LC_ADDR
LC_REQ
LC_ACK, LC_LACK
LC_DATA

FIG. 15

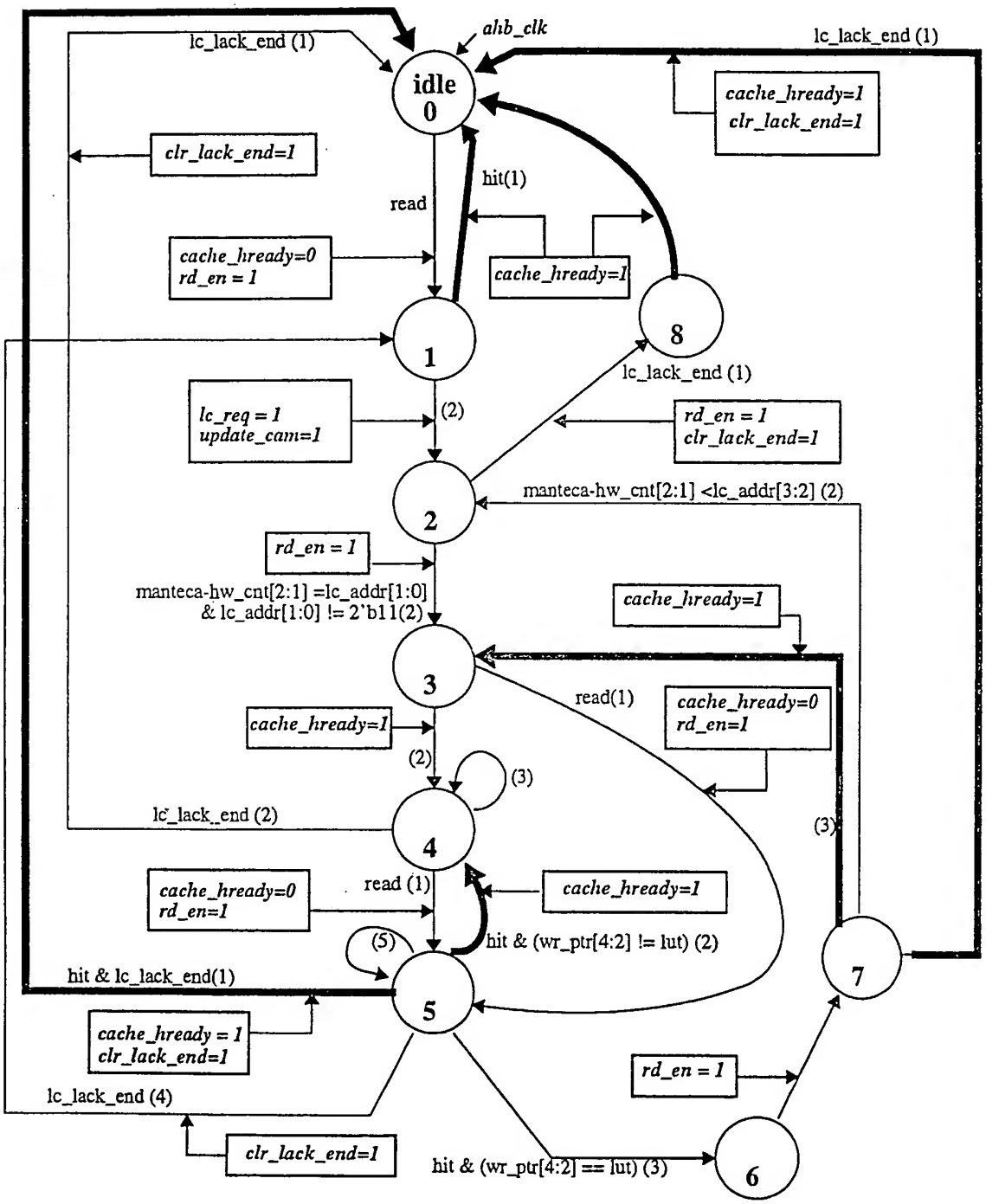
ahb_clk
lc_lack_end (1)
idle
0
clr_lack_end=1
read
hit(1)
cache_hready=0
rd_en = 1
cache_hready=1
1
lc_req = 1
update_cam=1
(2)
2
rd_en = 1
manteca-hw_cnt[2:1] =lc_addr[1:0]
& lc_addr[1:0] != 2'b11(2)
3
cache_hready=1
(2)
(3)
lc_lack_end (2)
4
cache_hready=0
rd_en=1
read (1)
(5)
hit & lc_lack_end(1)
5
cache_hready = 1
clr_lack_end=1
lc_lack_end (4)
clr_lack_end=1
hit & (wr_ptr[4:2] == lut) (3)
6
rd_en = 1
7
hit & (wr_ptr[4:2] != lut) (2)
cache_hready=1
read(1)
cache_hready=0
rd_en=1
(3)
8
lc_lack_end (1)
rd_en = 1
clr_lack_end=1
manteca-hw_cnt[2:1] <lc_addr[3:2] (2)
cache_hready=1
clr_lack_end=1
lc_lack_end (1)

FIG. 16

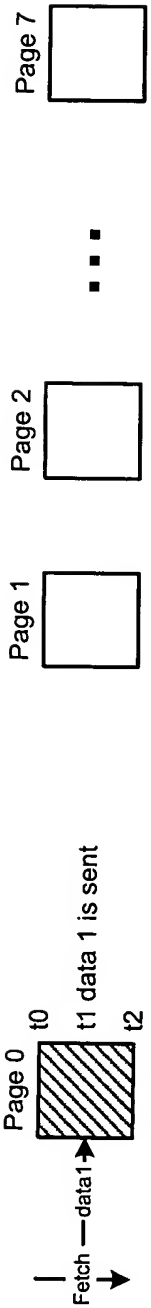

FIG. 17A

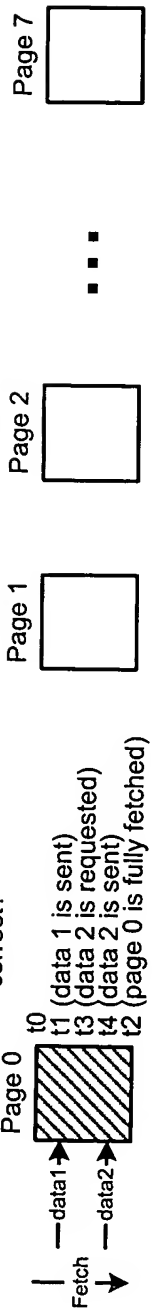

FIG. 17B

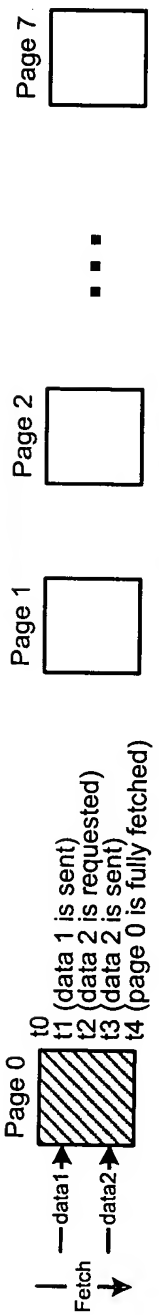

FIG. 17C

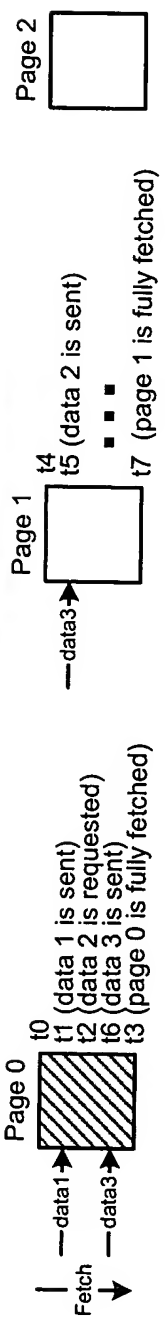

FIG. 17D

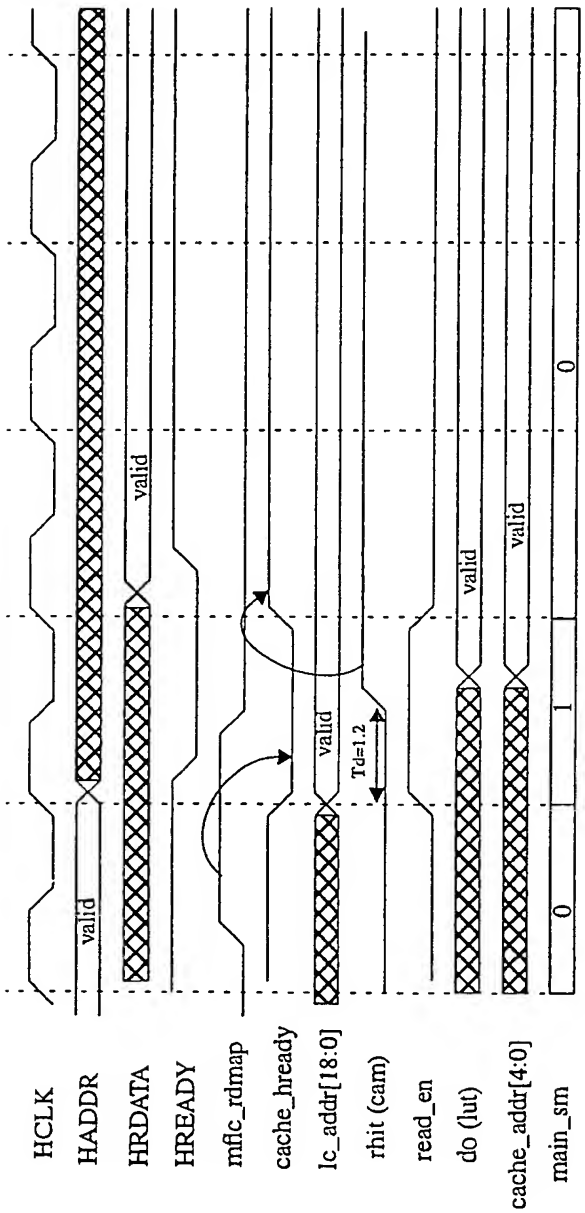
HCLK
HADDR
HRDATA
HREADY
mfic_rdmap
cache_hready
lc_addr[18:0]
rhit (cam)
read_en
do (lut)
cache_addr[4:0]
main_sm
valid
valid
valid
valid
valid
Td=1.2
0
1
0

FIG. 18

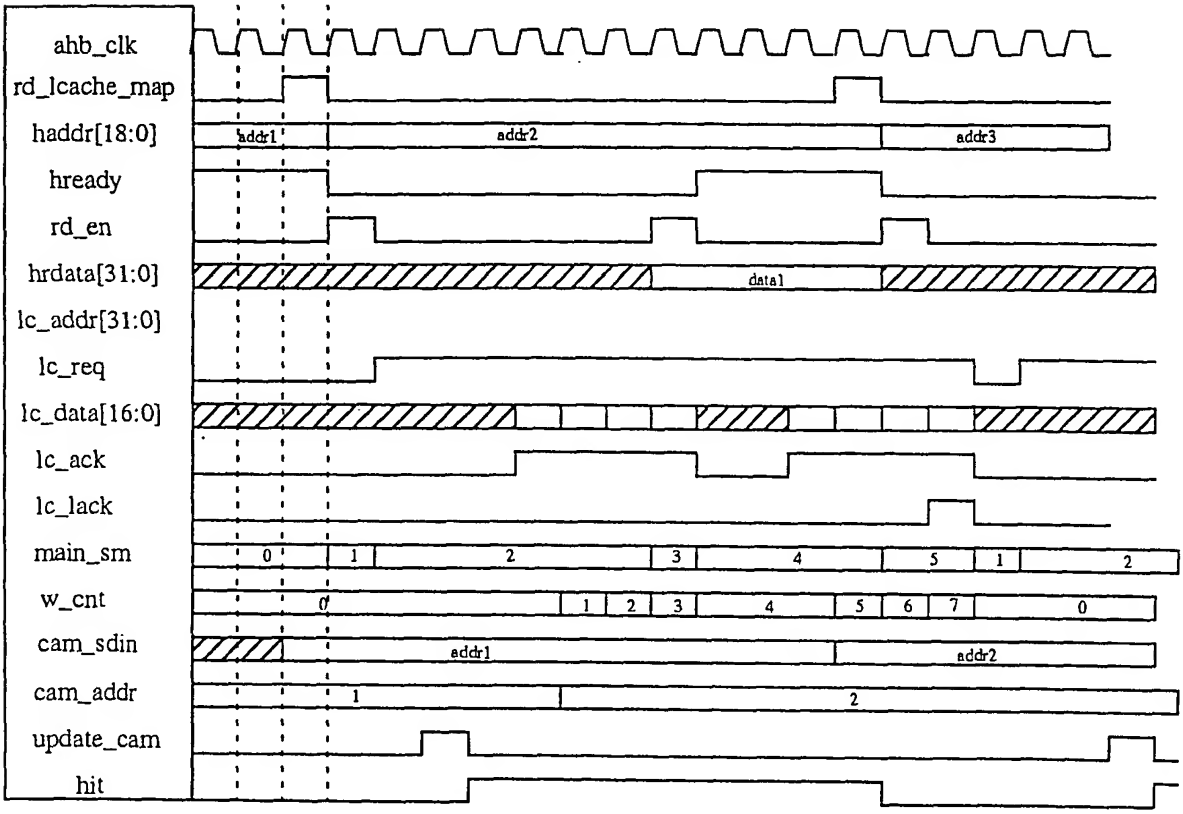
ahb_clk
rd_lcache_map
haddr[18:0]
hready
rd_en
hrdata[31:0]
lc_addr[31:0]
lc_req
lc_data[16:0]
lc_ack
lc_lack
main_sm
w_cnt
cam_sdin
cam_addr
update_cam
hit
addr1
addr2
addr3
data1
addr1
addr2

FIG. 19

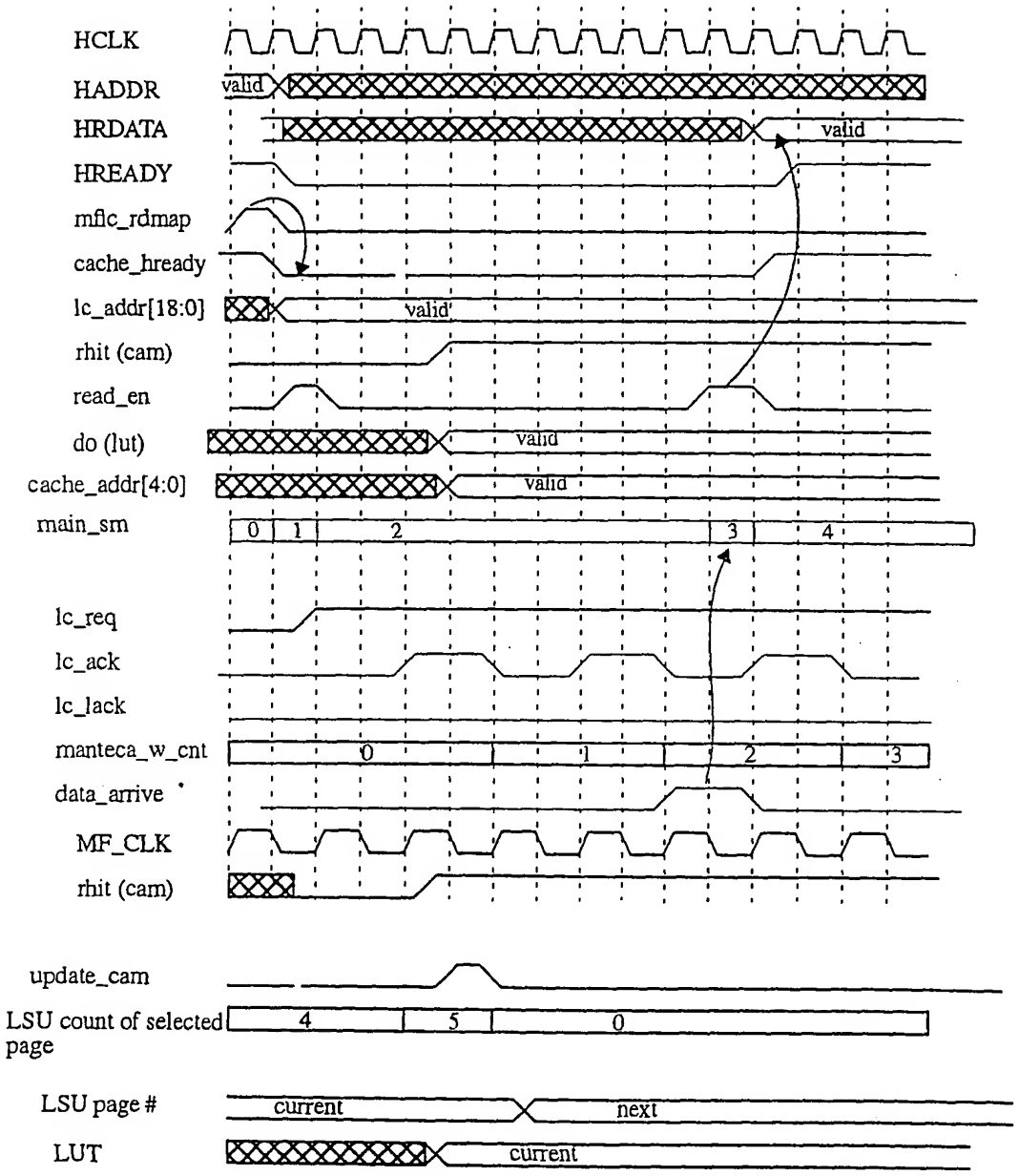
HCLK
HADDR
valid
HRDATA
valid
HREADY
mflc_rdmap
cache_hready
lc_addr[18:0]
valid
rhit (cam)
read_en
do (lut)
valid
cache_addr[4:0]
valid
main_sm
0
1
2
3
4
lc_req
lc_ack
lc_lack
manteca_w_cnt
0
1
2
3
data_arrive
MF_CLK
rhit (cam)
update_cam
LSU count of selected page
4
5
0
LSU page #
current
next
LUT
current

FIG. 20

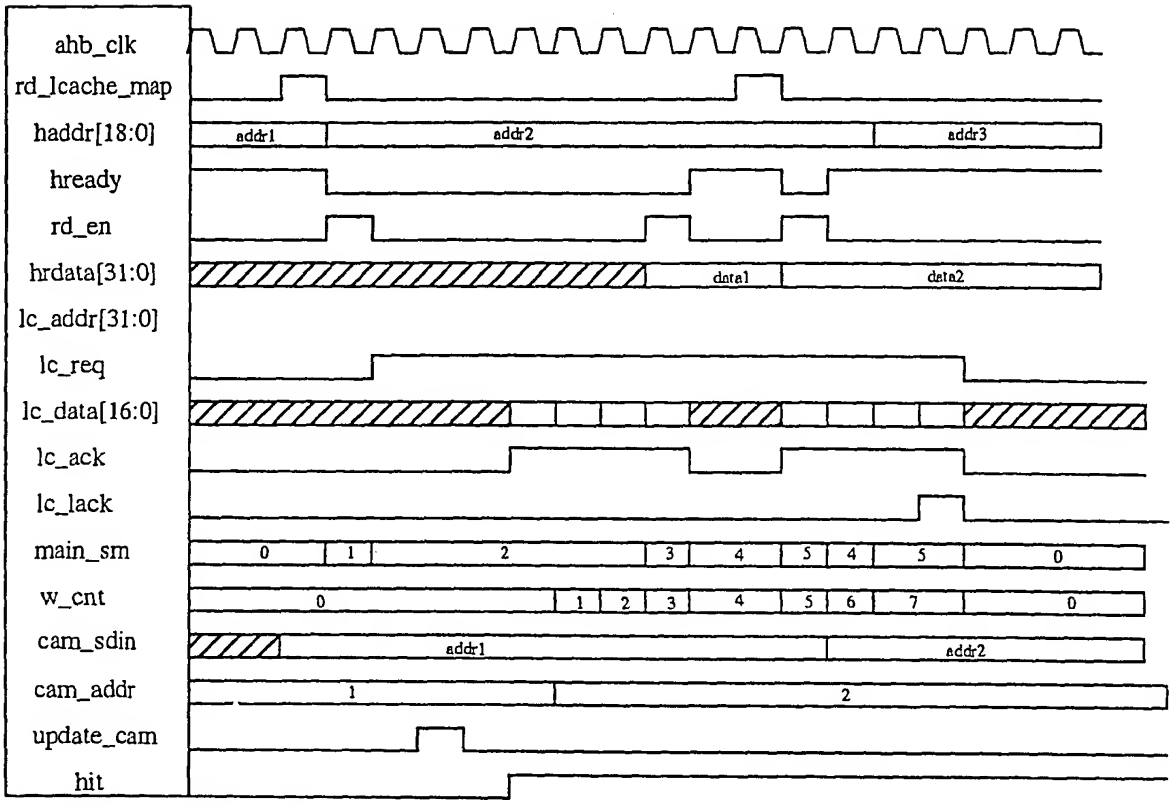
ahb_clk
rd_lcache_map
haddr[18:0]
addr1
addr2
addr3
hready
rd_en
hrdata[31:0]
data1
data2
lc_addr[31:0]
lc_req
lc_data[16:0]
lc_ack
lc_lack
main_sm
0 1 2 3 4 5 4 5 0
w_cnt
0 1 2 3 4 5 6 7 0
cam_sdin
addr1
addr2
cam_addr
1
2
update_cam
hit

FIG. 21

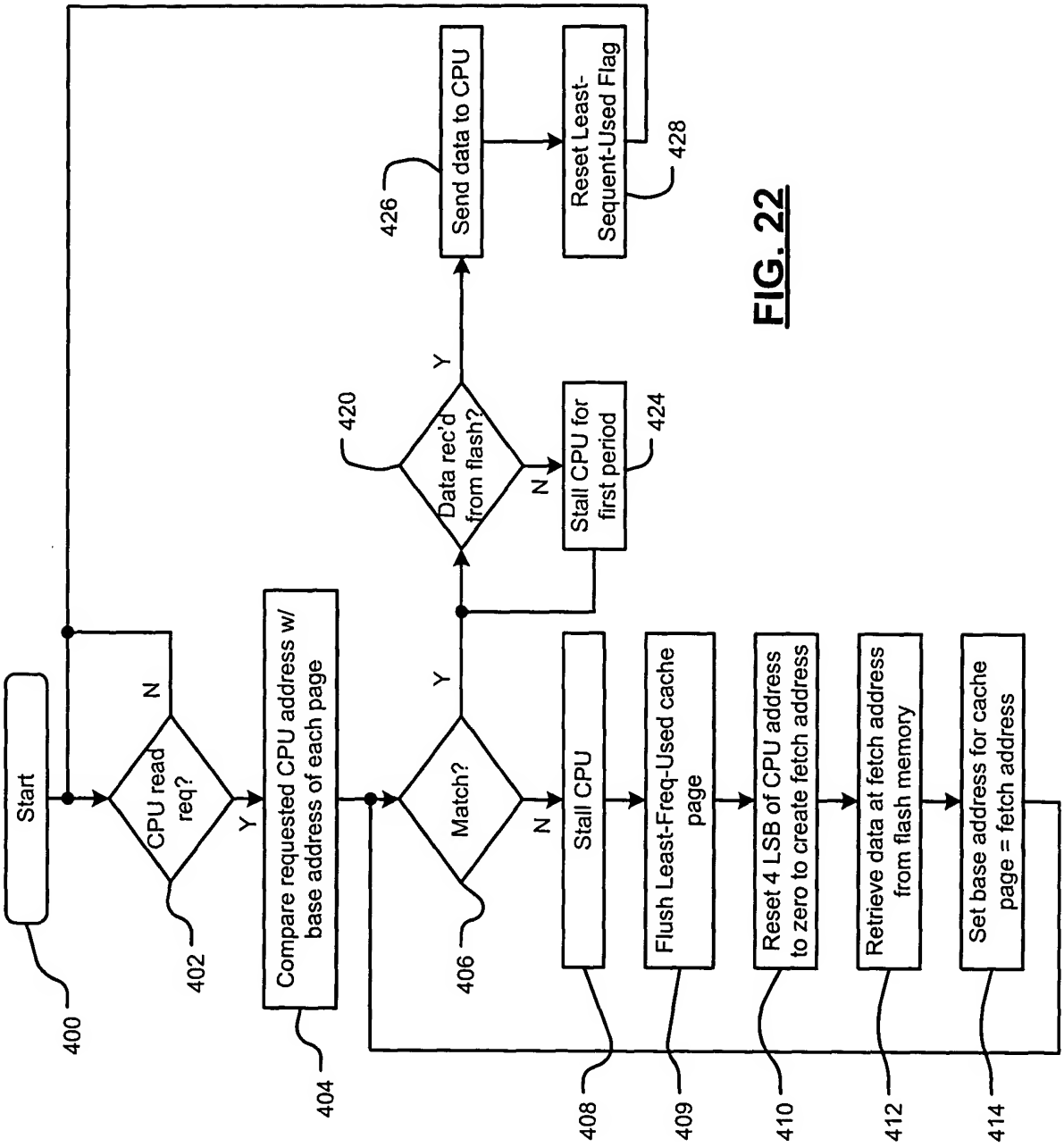
Start
400
CPU read req?
402
N
Y
Compare requested CPU address w/ base address of each page
404
Match?
406
Y
N
Stall CPU
408
Flush Least-Freq-Used cache page
409
Reset 4 LSB of CPU address to zero to create fetch address
410
Retrieve data at fetch address from flash memory
412
Set base address for cache page = fetch address
414
Data rec'd from flash?
420
Y
N
Stall CPU for first period
424
Send data to CPU
426
Reset Least-Sequent-Used Flag
428

FIG. 22

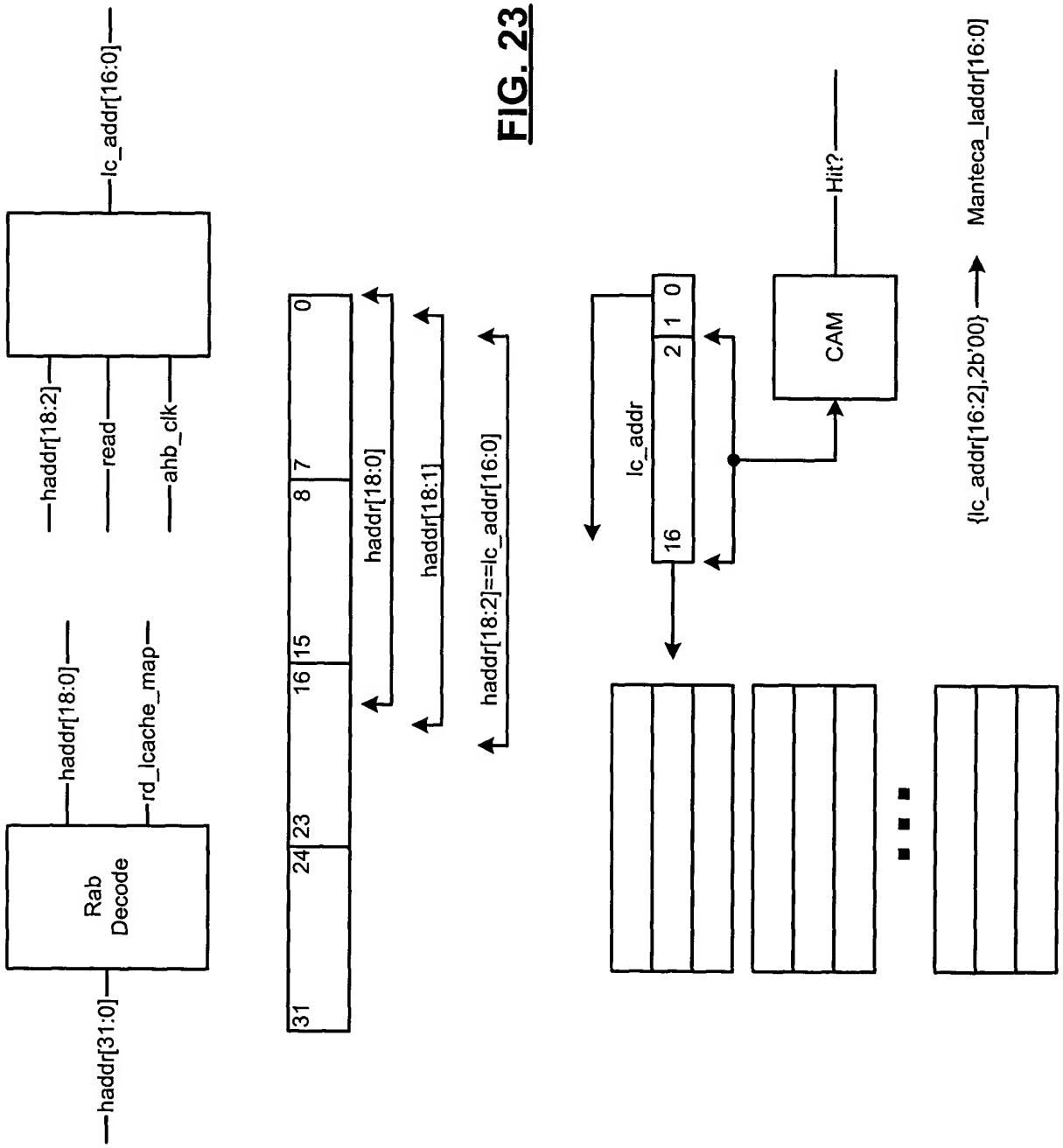
lc_addr[16:0]
haddr[18:2]
read
ahb_clk
haddr[18:0]
rd_lcache_map
Rab Decode
haddr[31:0]
31
24
23
16
15
8
7
0
haddr[18:0]
haddr[18:1]
haddr[18:2]==lc_addr[16:0]
lc_addr
16
2
1
0
CAM
Hit?
{lc_addr[16:2],2b'00}
Manteca_laddr[16:0]

FIG. 23